Volume 3, Issue 4, April 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Review on Protein function Extraction

Vinky *
Student Masters of Technology
Department of CSE
Sri Guru Granth Sahib World University
Fatehgarh Sahib, Punjab, India

Rajneet kaur
Assistant Professor & Head of Departmant of
Computer Science and Engineering
Sri Guru Granth Sahib World University
Fatehgarh Sahib, Punjab, India

Abstract - Genomics Research is growing rapidly as well as the information which is used to understand the latest discovery of protein functions, so it has become a tough task for the biomedical researchers to access this information. Various text mining techniques are used to fetch information from biomedical literature and transform that information to simple database formats. In this paper Sentence Pattern mining is used to extract protein functions from biomedical literature. These methods are used to support database managers in writing protein functions and to assist biologists and researchers in searching protein functions.

Keywords - Text mining, bioinformatics, knowledge acquisition, linguistic processing.

I. Introduction

'Protein Function' is an operational concept. Protein performs most important tasks in organisms like catalysis of Biochemical reactions, transport of nutrients, recognition and transmission of signals. Protein function is not a well defined term instead function is a complex phenomenon that is associated with many mutually overlapping levels (biomedical, cellular etc.). Understanding protein functions is the most basic and important goal. It is difficult for biomedical researchers to read and understand functions of proteins from volumes of papers in less time as these are not accessible from database. Therefore, text mining becomes an important technique to support database managers in facilitating the annotation process.

This paper deals with problem of protein function extraction from medical literature. The purpose is to extract 'Protein-GO-Document' relation.

Definition: If p is a protein having GO function g in a document d, then the triple (p, g, d) is called a protein-GO-document relation. Go stands for Gene Ontology is a biological ontology that provides a controlled vocabulary to describe knowledge of gene and protein roles in cells. In this paper, Sentence Pattern mining method is used which is combination of Phrase based pattern matching and sentence classification approaches.

II. Literature Extraction Framework

Fig. shows the Literature Extraction Framework. The input documents are full text articles or abstracts and the output is protein-GO-document relations.

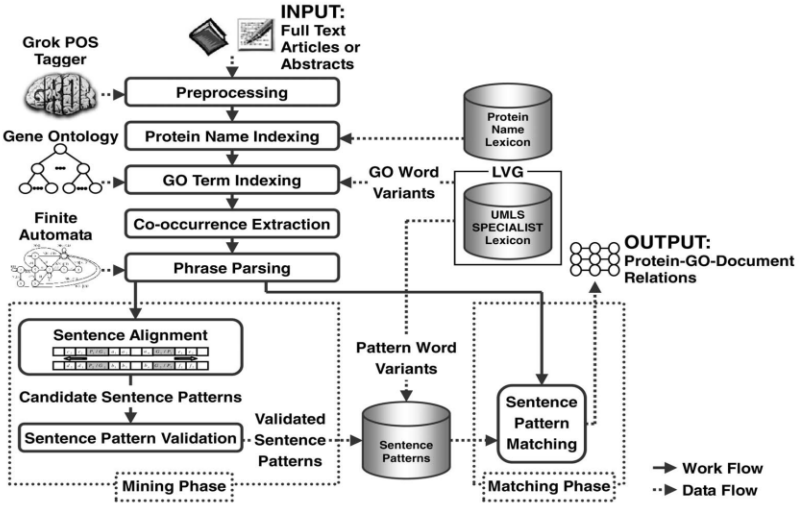

Fig. Literature Extraction Framework[1]

Input documents are processed through the steps of preprocessing, protein name indexing, GO term indexing, co-occurrence extraction and phrase parsing to transform sentences to phrase structures. Them, work low is divided into two phases:

Mining and Matching

In the Mining phase, sentence patterns are mined from sample sentences that describe protein functions,

In the Matching phase, these sentence patterns are then matched with new sentences to extract protein-GO-document relations.

The steps are as following:

A. Preprocessing

Sentences in the documents are detected and indexed by recording their positions in text for the purpose of returning the original text, and providing the evidence text of protein functions. These sentences are then tokenized.

B. Indexing of Protein names and GO terms

Indexing is used to identify protein names and GO terms in text. A word in text may match more than one protein name or GO term, even both, so tagging cannot be used as it has the limitation of overlapping names whereas indexing method records the positions of names in the database

Flexible pattern matching is performed to skip punctuation marks and extra parenthesis in sentences to recognize variations of protein names and GO terms. Indexing of GO terms is difficult so processing of GO term variants is done.

C. Recognition of GO terms variants

Variations of GO terms is performed to skip punctuation marks:

Morphological, Syntactic, Semantic

Morphological variants- One or more words of the original term are replaced with their morphologically related words in the variant and other words remain unchanged.

Syntactic variants- The content words of the original term are found in the variant, but the syntactic structure of the term is modified.

Semantic variants- One or more words of the original term are replaced with their synonyms in the variant and the other words remain unchanged.

Morphological variants are identified by adopting the Java Lexical tools, which uses the UMLS SPECIALIST Lexicon to handle lexical variants.

To identify syntactic variants, GO variation rules are mined from biomedical literature. The variation rule format is:

$$(X_i|Y_j) + \rightarrow (X_i|Y_k) +$$

Go term **Variant**

X_i is the token sequences that appear both in term and variant.

Y_j is the token sequences that appear only in term.

Y_k is the token sequences that appear in variant.

To deal with semantic variants, synonyms of GO terms are compiled.

D. Extraction of Co-occurrence Sentences

In this step, we extract sentences with the co-occurrences of a protein name and a GO term, which are called co-occurrence sentences by executing a SQL query.

Co-occurrence can be divided into 2 categories:

a. protein-GO, means protein name occurs first then followed by GO term

b. GO-protein

GO terms are divided into three categories:

Molecular function, biological process, and cellular component.

E. Phrase parsing

In sentences describing protein functions, there can be a lot of sentences that have similar formats but use different modifiers.

Shallow parsing method on co-occurrence sentences is used, as technique of full parsing sentences in medical documents is inefficient and prone to errors.

After parsing, a sentence is transformed into a phrase structure.

III. Techniques Used

There are various pattern mining methods. Some of them are:

Frequent pattern Mining- It finds a set of patterns that occur frequently in a data set, where a pattern can be a set of items, a subsequence, or a structure. A pattern is considered frequent if its count satisfies a minimum support.

Sequential Pattern Mining- It is the mining of frequently occurring ordered events or subsequences as patterns.

For example, A sequential pattern is "Customers who buy a Digital Camera are likely to buy a color printer within a month".

Sentence Pattern Mining- Given a set of co-occurrence sentences which have been transformed into parsed phrases, sentence pattern mining is to find the complete set of sentence patterns in the set of sentences.
In this paper, Sentence Pattern Mining is used.
In sentences that report protein functions, we can find many sentence patterns i.e. wording or writing styles used by authors to describe protein functions.
For Example, "<protein> participates in <GO>" and "<protein> is localized to <GO>"

These sentence patterns are very useful characteristics for identifying sentences describing protein functions.

A sentence pattern:
$SP = \{C_{prefix}E_1C_{infix}E_2C_{suffix}\}$
is a sequence of parsed phrases. E_1, E_2 are parsed phrases, which represent named entities to be extracted.
P, G is protein names and GO-term respectively.
C_{prefix}, C_{infix}, C_{suffix} are sequences of parsed phrases which represent contextual phrases of the named entities.

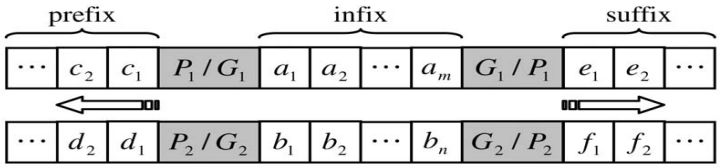

Fig. Sentence alignment between a pair of co-occurrence sentences. The arrows indicate the directions of matching in the prefix and suffix parts of the pair of sentences.[1]

For sentence pattern mining, co-occurrence sentences are divided into positive and negative examples.
Positive examples are co-occurrence sentences where occurring GO-function in database annotation and other are negative examples.
Sentence Pattern Mining consists of 3 steps:
(i) Candidate sentence patterns are mined from positive examples by aligning each pair of sentences.
(ii) The support and confidence of each candidate sentence pattern is calculated by matching the pattern with each positive or negative example.
(iii) Candidate sentence patterns are screened according to their support and confidence levels, in order to acquire appropriate sentence patterns.

Algorithm: SentencePatternMining (E^+)
/* This is the main algorithm.*/
Input: a set of positive examples E^+
Output: a set of candidate sentence patterns C
1: **begin**
2: **for** each pair of positive examples s_i, $s_j \in E^+$ **do**
3: **begin**
4: // The same sentence cannot be aligned.
5: **if** (s_i.sentenceID $\neq$ s_j.sentenceID) **then**
6: **begin**
7: p = AlignExample(s_i, s_j)
8: if (p $\neq$ null ^ p $\notin$ C)
9: Add p to C
10: **end**
11: **end**
12: **end**

Procedure 1. AlignExample(s_i, s_j)
/* This is the procedure called by SentencePatternMining().*/
Input: a pair of positive examples s_i and s_j
Output: a candidate sentence pattern p
1: **begin**
2: **if**(s_i.infix.phraseNum $\neq$ s_j.infix.phraseNum) **then**
3: **return** null
4: **for** each pair of corresponding phrases $h_k \in s_i$.infix and $h_l \in s_j$.infix **do**
5: **if**(¬ AlignPhrase (h_k, h_l)) **then return** null
6: **for** each pair of corresponding phrases $h_k \in s_i$.prefix and $h_l \in s_j$.prefix, from right to left, **do**


```
7:        if( ¬ AlignPhrase ( h_k, h_l)) then break
8:    for each pair of corresponding phrases h_k ∈ s_i.suffix and h_l ∈ s_j.suffix, from left to right, do
9:        if( ¬ AlignPhrase ( h_k, h_l)) then break
10:   Create a pattern p from the matching phrases and tokens in s_i
11:  return p
12: end
```

Procedure 2. AlignPhrase (h_k, h_l)

```
/* This is the procedure called by AlignExample (). */
```

Input: a pair of phrases h_k and h_j

Output: true (h_k matches h_l) or false

```
1: begin
2:   start =0
3:   match =0
4:   for i =0 to h_k.tokenNum do
5:     for j =start to h_k.tokenNum do
6:       if(AlignToken(h_k.token[i] , h_l.token[j])) then
7:       begin
8:          start= j+1
9:          match= match+1
10:         break
11:       end
12:   if(match= 0) then return false
13:   else return true
14: end
```

Procedure 3. AlignToken(t_i, t_j)

```
/* This is the procedure called by AlignPhrase (). */
```

Input: a pair of tokens t_i and t_j

Output: true (t_i matches t_j) or false

```
1: begin
2:   for each pair of token variants v_k ∈ t_i.variantSet and v_l ∈ t_j.variantSet do
3:      if (v_k= v_l)
4:      begin
5:         t_i.matching = true
6:         return true
7:      end
8: return false
9: end
```

To check the effectiveness of our methods the performance of GO-term indexing, protein-name indexing and sentence pattern mining is evaluated by calculating their support and confidence.

$$\text{support (p)} = \text{positive (p)} / N$$

$$\text{confidence (p)} = \text{positive (p)} / \text{positive (p)} + \text{negative (p)}$$

IV. Conclusion

This paper presents a new methodology for extracting protein functions from biomedical literature. In this paper Sentence Pattern Mining method is used which is combination of Phrase based pattern matching and sentence classification approaches. Pattern Mining can reduce the effect of pattern construction. By this method description of protein functions in articles is recognized. GO-term variants improve the performance of GO-term indexing. This study facilitates the understanding of protein functions for biologist and medical researchers.

References

[1] Jung-Hsien Chiang and Hsu-Chun Yu, "Literature Extraction of Protein Functions Using Sentence Pattern Mining", IEEE Transactions on knowledge and data engineering, vol. 17, no. 8, pp. 1088-1098, 2005.

[2] A. Bairoch, B. Boeckmann, S. Ferro, and E. Gasteiger, "Swiss-Prot: Juggling between Evolution and Stability," Briefings in Bioinformatics, vol. 5, no. 1, pp. 39-55, Mar. 2004.

[3] E. Camon, D. Barrell, V. Lee, E. Dimmer, and R. Apweiler, "The Gene Ontology Annotation (GOA) Database—An Integrated Resource of GO Annotations to the UniProt Knowledgebase," Silico Biology, vol. 4, no. 1, pp. 5-6, 2003.

[4] J.-H. Chiang and H.-C. Yu, "MeKE: Discovering the Functions of Gene Products from Biomedical Literature via Sentence Alignment, "Bioinformatics, vol. 19, no. 11, pp. 1417-1422, 2003.

[5] J.-H. Chiang, H.-C. Yu, and H.-J. Hsu, "GIS: A Biomedical Text-Mining System for Gene Information Discovery," Bioinformatics, vol. 20, no. 1, pp. 120-121, 2004.
[6] N. Daraselia, A. Yuryev, S. Egorov, S. Novichkova, A. Nikitin, and I. Mazo, "Extracting Human Protein Interactions from MEDLINE Using a Full-Sentence Parser," Bioinformatics, vol. 20, no. 5, pp. 604-611, 2004.
[7] Grok, http://grok.sourceforge.net/, 2012.
[8] W. Hersh and R.T. Bhupatiraju, "TREC Genomics Track Overview,"Proc. 12th Text Retrieval Conf. (TREC 2003), 2003, http://trec.nist.gov/pubs/trec12/t12_proceedings.html.
[9] L. Hirschman, J.C. Park, J. Tsujii, L. Wong, and C.H. Wu,"Accomplishments and Challenges in Literature Data Mining for Biology," Bioinformatics, vol. 18, no. 12, pp. 1553-1561, 2002.
[10] C. Jacquemin, Spotting and Discovering Terms through NLP. Cambridge, Mass.: MIT Press, 2001.
[11] G. Leroy and H. Chen, "Filling Preposition-Based Templates to Capture Information from Medical Abstracts," Proc. Pacific Symp. Biocomputing (PSB) 2002, pp. 350-361, 2002.
[12] A.T. McCray, S. Srinivasan, and A.C. Browne, "Lexical Methods for Managing Variation in Biomedical Terminologies," Proc. 18th Symp. Computer Applications in Medical Care (SCAMC '94), pp. 235-239, 1994, http://umlslex.nlm.nih.gov/.
[13] C. Perez-Iratxeta, P. Bork, M.A. Andrade, "Exploring MEDLINE Abstracts with XplorMed," Drugs Today, vol. 38, no. 6, pp. 381-389, 2002.
[14] J. Pustejovsky, J. Castan˜ o, J. Zhang, M. Kotecki, and B. Cochran, "Robust Relational Parsing over Biomedical Literature: Extracting Inhibit Relations," Proc. Pacific Symp. Biocomputing (PSB) 2002,pp. 362-373, 2002.
[15] S. Raychaudhuri, J.T. Chang, F. Imam, and R.B. Altman, "The Computational Analysis of Scientific Literature to Define and Recognize Gene Expression Clusters," Nucleic Acids Research, vol. 31, no. 15, pp. 4553-4560, 2003.
[16] M. Sipser, Introduction to the Theory of Computation. Boston: PWS, 1997.
[17] B.J. Stapley, L.A. Kelley, and M.J.E. Sternberg, "Predicting the Sub-Cellular Location of Proteins from Text Using Support Vector Machines," Proc. Pacific Symp. Biocomputing (PSB) 2002, pp. 374-385, 2002.
[18] L. Tanabe and W.J. Wilbur, "Tagging Gene and Protein Names in Biomedical Text," Bioinformatics, vol. 18, no. 8, pp. 1124-1132, 2002.

